HistoTrust : Tracing AI behavior with secure hardware and blockchain technology

Dylan Paulin, Raphaël Joud, Christine Hennebert, Pierre-Alain Moellic, Thibault Franco-Rondisson, Romain Jayles

20

HistoTrust : Tracing AI Behavior with Secure Hardware and Blockchain Technology

Dylan Paulin[1*], Raphaël Joud[1], Christine Hennebert[1*], Pierre-Alain Moëllic[1], Thibault Franco-Rondisson[1] and Romain Jayles[1]

[1*]Univ. Grenoble Alpes, CEA-Leti, Grenoble, France.

*Corresponding author(s). E-mail(s): dylan.paulin@cea.fr; christine.hennebert@cea.fr;
Contributing authors: raphael.joud@cea.fr; pierre-alain.moellic@cea.fr; thibault.franco-rondisson@cea.fr; romain.jayles@cea.fr;

Abstract

In areas of activity where the notion of accountability is strong, the adoption of artificial intelligence is limited by the opacity and lack of understanding of its behavior. All the more so in the embedded domain where neural networks are compressed and executed on microcontrollers. While the NIST introduced in 2021 several principles allowing the AI explainability, this paper introduces a novel scheme, HistoTrust, combining secure hardware and blockchain technology to bring trust in the traceability of AI behavior and allow its explainability. HistoTrust attests in an ethereum ledger all the relevant data produced by a physical device, especially the heuristics inferred by AI. Thus, the audition of the ledger enables security verifications and AI behavior analysis.

Keywords: hardware security, blockchain technology, attestation scheme, embedded neural network, AI explainability

1 Introduction

From the perspective of the factory of the future, smart robots are increasingly incorporating vision capabilities based on an on-board camera. From the

pictures, embedded artificial intelligences (AI) make decisions impacting the tasks performed by the robot within the industrial process. The AI is previously trained to recognise learnt patterns in the image. The classifier built is a neural network (NN), which given an image as input, infers a probability for the recognition of the learnt pattern. A high probability provides trust in the recognition of the learnt pattern. With AI, this trust is based on a probabilistic process.

The adoption of AI in the industry is being slowed down by the opacity of the decision making when an IA is involved in the decision process. That's why in september 2021, the NIST published the report [1] that promulgates four principles to enable the AI explainabilibity. Among these principles, the transparency of the AI behavior is a key factor of trust along with accountability and resiliency.

When an anomaly is detected on a production line, the causes and accountabilities must be determined. When the production process involves AIs, implementing the means to trace events and audit the digital system is a requirement. The solution Histotrust [2] aims to provide such a tool to ensure the protection of embedded AIs against malicious intends and to enable the explainability of the AIs behavior. Histotrust combines the probabilistic trust provided by AI with the deterministic trust provided by the blockchain. The notion of trust in the blockchain is based on a consensus protocol between the actors involved, enabling them to agree on the transactions recorded in the ledger [4]. Once recorded, the transactions form an history considered as immutable. They can no longer be deleted, swapped or modified. Also, the integrity of the information recorded in the ledger is ensured by design, as well as the ordering of events and the authentication of issuers. The blockchain technology is relevant to trace, in a non-repudiable way, the activity of smart robots, and embedded NN.

Histotrust introduces a device-centric [5] solution based on Ethereum technology that conciliates the need for security and privacy with the trust required between stakeholders. HistoTrust provides an architecture that ensures end-to-end security and privacy by design while enabling the traceability of embedded NN inferences. The authenticity of the issuer device is attested through secure hardware components such as Trusted Platform Module (TPM) and ARM TrustZone technology as Trusted Execution Environment (TEE). Hardware components serves as root-of-trust for the digital data processed by the embedded NN.

Thus, each of the smart robots operating on the production line sends to the ledger the attestations of the digital data it produce. An attestation includes the cryptographic fingerprint of a set of raw data, the authentication of the issuing embedded applications, and the timestamp of the record. The ledger maintains the history of transactions received from the smart robots distributed around the production line. In a context where several stakeholders cooperate in the manufacture of a product, each protecting its own interests, business and personal data, sharing attestations through the ledger brings

trust between them. While each one keeps and protects its raw data, and must be able to explain the behaviour of its embedded AI if requested.

Histotrust has several objectives: 1) to protect the embedded NN from logical and physical attacks by ensuring the cyber robustness of the AI, 2) to protect the data produced by the embedded applications and processed by the NN in order to allow the explainability of the AI behavior, 3) to attest and trace the data produced in a blockchain in order to provide authentic non-repudiable attestations shared between the different stakeholders.

The following section positions the work done in HistoTrust in relation to existing solutions. The use case is described in section 3. Section 4 presents the embedded NN used in Histotrust. Section 5 outlines the attestation process of the data produced to the ledger.The integration with the embedded NN and the deployment is discussed in section 6. A security analysis is led in section 7 following by the audition process in section 8 before concluding this work.

2 Related works

2.1 Secure data history with trusted hardware

The added value of blockchain technology to meet the specific features of a smart manufacturing use case has been shown in [6]. Compared to a centralized solution based on digital certificates and PKI, the Ethereum-based solution offers a more refined management of security and privacy at the expense of performance. In [2], HistoTrust demonstrates that performance can met the needs of a real-time usage when using a blockchain.

The EmLog framework [7] is presented as *"the first attempt at preserving off-the-shelf ARM development board hosting OP-TEE"*. EmLog implements a secure logging system from end-to-end between embedded constraint devices and a remote database. HistoTrust introduces an architecture design and an on-board implementation design using off-the-shelf secure hardware components, as OP-TEE and TPM 2.0 [8], that goes beyond EmLog solution and achieves the EmLog perspectives. Preserving forward security thanks to the one-way hash chain scheme introduced by Shneier and Kelsey [9], EmLog and SGX-Log [10] are not designed for multi-stakeholders contexts and may suffer of data losses in case of power failure.

In the Logs system EngraveChain detailed in the paper [11], the data history is ciphered, then registered in an Hyperledger Fabric ledger. This implementation lacks agility because the blockchain is not designed to store large volumes of data, nor confidential data even encrypted. Moreover, the ciphering of recorded data in a ledger implies a complex key management. The blockchain technology provides by design the tamper-resistance of the recorded transactions history forming the ledger. HistoTrust provides an attestation scheme securing the history of data issued from distributed devices.

An Ethereum ledger maintains the history of cryptographic attestations of data produced by distributed devices owning by multiple stakeholders. The blockchain technology enables to share these cryptographic evidences between

the stakeholders providing trust. In addition, the raw data is kept by their owners who ensure their persistence and confidentiality.

Based on an Ethereum blockchain, BlockPro [12] presents a decentralised architecture of IoT devices. The authenticity of the devices issuing data is achieved through a challenge to the IoT device submitted to its PUF (Physical Unclonable Function). But it is not mentioned how the account address issuing the transactions is built and how it is linked to the PUF. Paper [13] shows that dissociating IoT devices and validator nodes is a powerful architecture that HistoTrust exploits.

2.2 Attestation scheme

Attestation schemes based on the use of a TPM offer standard solutions allowing the authentication of a platform by a remote device [14] [15]. The authors of [16] highlight the question of the certification of sensor data, even by a trusted platform. The tension between privacy, which requires the protection of confidential data, and trust, which requires guarantees between the stakeholders working in a given ecosystem is tangible.

The principle of remote attestation is described in depth in [15]. The Trusted Platform Module (TPM) is the targeted device enabling the endorsement of attestation keys that the manufacturer, the vendor or the owner may own. The attestation scheme follows recommendations and standards provided by the Trusted Computing Group (TCG) [14]. Attestation aims at proving to a remote verifier the property of a target by supplying an evidence over a network. It consists in three stages: 1) key provisioning, 2) attestation process, 3) verification process.

2.3 Explainability of embedded artificial intelligence

The field of Explainable AI (XAI) raises major attention as an important concept that increases the trust in AI-based systems and applications. The need of both interpretability and explanation methods has been recently highlighted by the NIST [1]. A large variety of approaches have been proposed to enlighten the blackbox paradigm of deep NN models [17] even for modern architectures.

The purpose of our work is not to introduce a new methodology to explain the intrinsic behavior of a Machine Learning (ML) model, but to frame the implementation of an AI in an embedded device in such a way that one can trust the confidential data, presented to a third party, to explain the behavior of an embedded NN. Our contribution is rather in the area of cyber robustness of embedded AI in the presence of multiple distributed NNs.

3 Use Case

3.1 Context

In a factory, many actuators participate in the assembly of a product on a production line (see figure 1). Physical devices that embed inference engines,

i.e. a NN previously trained to recognise determined patterns in an image, generate the digital commands sent to the actuators. The device may integrate several sensors and a camera. A picture of the product is taken before acting. This picture is presented in input of the NN to request an inference that contains heuristics, i.e. probabilities that the pattern recognised in the image corresponds to the learned patterns. This inference will guide the decision about the next action the actuator should perform.

When an incident occurs, the causes and the accountabilities should be determined. However, the presence of AI makes difficult the reproduction of the decisions. So, how to determine who is accountable for the damage? In particular, who is accountable for the decisions that command the actuators? If the NN recognises the digit '2' with a higher probability than the digit '8', whereas the digit is '8', is the error attributable to the learning quality? A configuration and/or system integration fault? A lack of operator guidance? Noisy input data? A physical or logical attack on the electronic devices? A network attack?

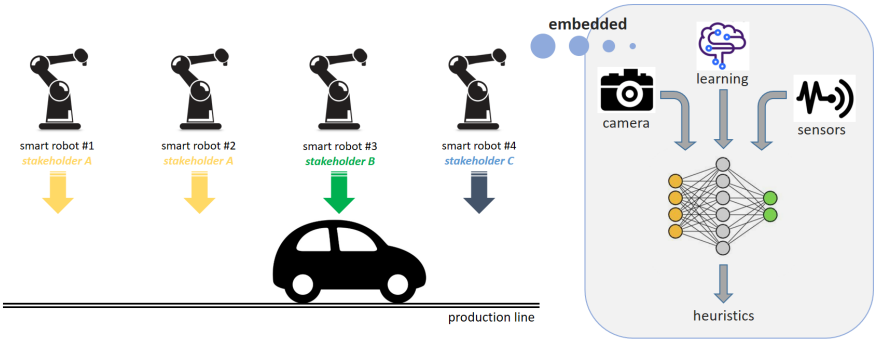


Fig. 1: Illustration of the use-case

3.2 Digit recognition

Smart robots are often equipped with cameras that allow them to photograph the part of the product on which they will operate. The image is then analysed, potentially with a classifier, and depending on the patterns recognised, the action is determined. For this work, we use a classical digit recognition task with the MNIST dataset [18] as it represents one of the most popular benchmarks in the ML literature with which many architectures can be tested (from shallow fully-connected networks to deeper convolutional NN). MNIST is composed of 60,000 training images of gray-scale handwritten digits and 10,000 examples for test. Each sample is a grayscale 28x28 image (784 pixels) with the associated label '0' from '9'. This data set offers a school case with a known and qualified opensource model. The integration made for the use case

can be generalised to other computer vision tasks, specific to the problem to solve.

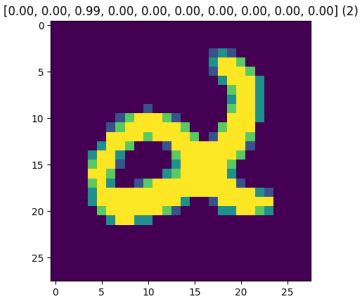


Fig. 2: Recognition of the digit 2

For a given picture in input of the NN, the output inference is composed of 10 heuristics that correspond to the probability of recognition of each digit from '0' to '9'. An example is shown in the figure 2 with the recognition of the digit '2' with the probability $0,99$ (99

4 Embedded AI

4.1 Formalism

In this work, we consider a deep NN model that performs a supervised classification task with the following formalism. Input-label pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$ are sampled from a distribution $\mathcal{D}$. The NN model $M_\Theta : \mathcal{X} \to \mathcal{Y}$, with parameters Θ, classifies an input $x \in \mathcal{X}$ to a label $M_\Theta(x) \in \mathcal{Y}$. The parameters are optimized during the training phase in order to minimize a *loss* function $\mathcal{L}\big(M_\Theta(x), y\big)$ (e.g., the cross-entropy loss) that evaluates the quality of a prediction compared to the ground-truth label. For the sake of readability, the model M_Θ is simply noted as M.

We distinguish a model, M, as an *abstract algorithm* from its *physical implementations* M. One model M (e.g., a CNN trained on MNIST for digit recognition) can be implemented for inference purpose in a microcontroller or in FPGA. Functionally, the embedded models rely on the same abstraction M but strongly differ in terms of implementation along with their respective hardware environments. Thus, there is no equivalence between M and its embedded variants.

Embed deep NN models on a constrained platform such as a 32 bits microcontroller usually needs model compression techniques to fit the model complexity to the hardware requirements [19]. More particularly, memory footprint is usually an important challenge: for a typical Cortex-M MCU, the trained parameters are stored in the Flash memory and, at inference time,

the internal computations (mainly multiply-accumulations and non-linear activations) are processed in SRAM. Two classical approaches are used to fit state-of-the-art models: *quantization* and *pruning*. Although the learning process may require 32 bits floating point computations, at inference time, a low bitwidth representation of the parameters is sufficient and does not alter the performance of the model. Thus, most of the tools that enable NN embedding on MCU (such as STM32Cube.MX AI[1]) propose a 8-bit quantization of the parameters. Pruning refers to techniques that *cut* useless connections in the network and rely on the fact that most of the models are over-parametrized. Both approaches can also help speeding up the inference process.

4.2 Neural network

Two different architectures of model working on MNIST dataset have been used, a MLP and a CNN. Both needed to be small to fit hardware material limitations. As such, MLP is composed of an input (784 points due to the fact that the images must be flattened to be used) and an output layer (10 neurons corresponding to number of label). This model has only 7850 trainable parameters which makes it a quite small model compared to others doing same task with additional intermediate hidden layers. However, model accuracy is just below 92%. Despite that state-of-the-art MLP model can reach higher accuracy on MNIST classification, this accuracy remains acceptable in light of model reduced architecture.

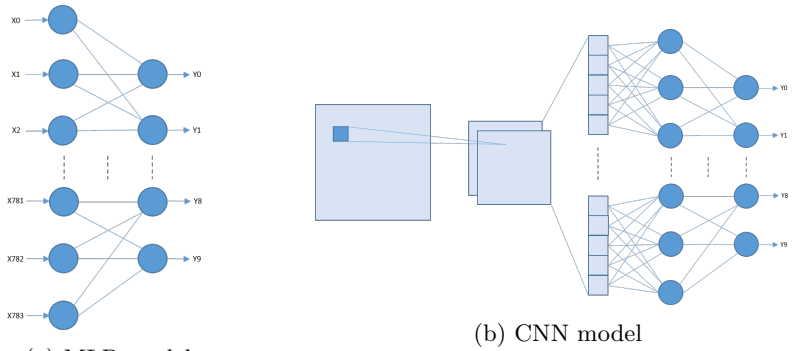
On the other hand, a CNN is also considered. This kind of model is divided in two parts with distinct goals. First layers and made for feature extraction (convolution, max pooling layers etc...) whereas end layers generally are regular MLP with dense layers.

CNN are particularly efficient and adapted for image recognition and classification as shown in the figure 3. Indeed, despite its reduced size, model reaches accuracy slightly over 96% for MNIST image classification.

4.3 Learning

In order to implement deep NN models on microcontrollers such as STM32, we previously generate the model with Google Tensorflow [20]. The model architecture (number of neurons, layers, used activation functions) is created according to the target specification, an ARM Cortex M4. Then, an *empty* model is trained with labelled data corresponding to the task to perform, the digit recognition, following a supervised learning paradigm. Validation and test of the dataset complete the training. The validation adjusts the hyper-parameters value and distinguish overfeating. The test qualifies the model performance with examples that have not been seen during the training phase. This allows the simulation of real model behavior while having ground truth class for each example of the dataset. At the end, TensorFlow provides an

[1] https://www.st.com/en/embedded-software/x-cube-ai.html



(a) MLP model

(b) CNN model

Fig. 3: Correlation score obtained by trace's samples at research cycle 0 for 10 best hypothesis

accuracy score. The trained model characteristics (architecture, parameters and hyper-parameters values) composes the embedded NN in a '.h5' file.

5 Attestations to ledger

The attestation scheme follows the 3 phases depicted in the figure 4:

1. The secrets and the trusted applications (apps) are provisioned in the embedded device by the device's owner in its private office. Once the secrets protected by secure hardware, the device is delivered in the factory.
2. In the factory, during the execution, the device is supervised by an operator. It produces data attested by a trusted app to a distributed ledger.
3. Any stakeholder may perform the verification of the authenticity of the involved devices, thanks to the information registered in the shared ledger, available to all. An accredited and independent auditor may also verify the tamper-resistance of the data produced.

5.1 Provisioning

5.1.1 Provisioning of the secret keys

The goal is to provision the private key *sk* in the TPM2 vault, while enabling its secure access from the TrustZone for the attestation phase, and the verification of its authenticity for the verification phase.

So, the private key *sk* is created by the device's owner in a private location. *sk* should have a high entropy and be on the elliptic curve secp256k1. To endorse *sk*, the owner generates *sk* certificate signed with its owner's master key *ok*. Previously, the owner has created its owner's master key ok, that may be supported in a PKI. Both owner master key *ok* certificate and endorsed device key *sk* certificate are in the ledger and available to all the stakeholders.

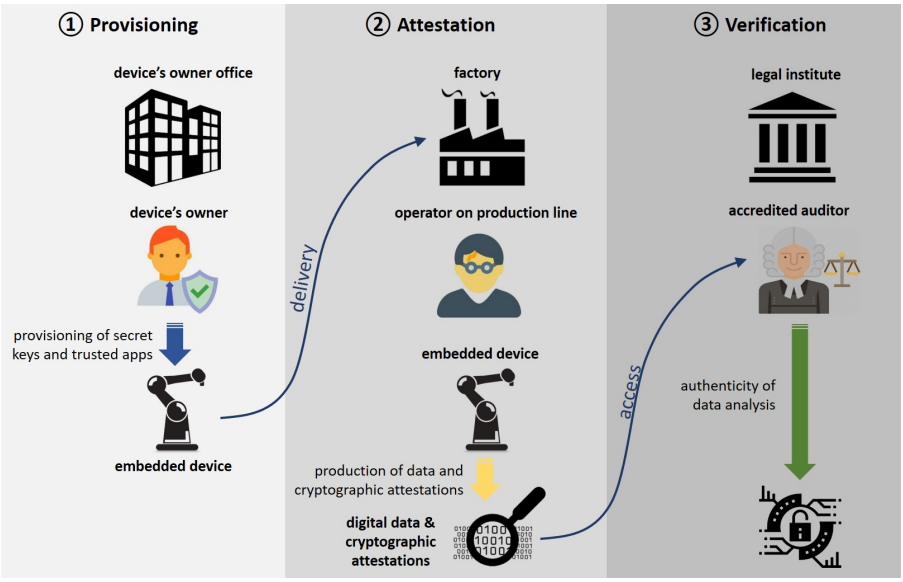


Fig. 4: the 3 phases of the attestation process

To avoid the eavesdropping of sk when it is accessed from the TrustZone, sk is ciphered with a symmetric key noted $symKey$. Once ciphered, the key sk_c is writen in the TPM permanent memory. The symmetric key symKey is also hidden in TrustZone, in order to decipher sk_c in a TEE when used.

5.1.2 Provisioning of the trusted apps

The ethereum technology requires that the incoming transactions are signed with a private key of the elliptic curve family secp256k1. However, this asymmetric cryptosystem is not supported by the TPM 2.0 standard and is not integrated in the TPM crypto-accelerator. That's why, for Histotrust, the cryptographic functions, dedicated to the compliance with ethereum technology, are implemented in TrustZone of an ARM microcontroller.

Two trusted apps are developed in Histotrust:*industrial app*: This application is the "business" application as it realizes the task required. It produces digital data that may be a huge value. *attestation app*: This application builds the cryptographic elements included in the transactions sent to the ethereum blockchain to attest the data produced.

The attestation app is composed of a part executed in the normal world of the microprocessor, and another part protected during the execution in the TrustZone. In order to carry out the measurement process 6.2, a fingerprint of the binary code of each app is computed and stored in the TPM PCR registry.

5.2 Attestation of the data produced

During the production phase, the cryptographic attestations are registered in Ethereum ledger through a smart contract. The attestation process, detailed in figure 5, consists in computing the fingerprint of the latter data set produced, that is included in the *data* field of an ethereum transaction 10. This transaction is signed in the TrustZone with sk which is also used to build the account address of the issuer device. To achieve the signature, the private key sk_c is accessed in the TPM permanent memory through the SPI bus and is deciphered in the TrustZone. The signed transaction is sent to the blockchain and a receipt is returned if the registration in the ledger is confirmed. The implementation of this attestation process is tricky because it must respect several temporal constraints, while following the real-time of industrial app that produces new data. No data should be lost, either because of the processing time of the attestation app, or a power failure of the physical device, or the latency of recording in the remote blockchain. In fact, the use of secure hardware components, as TPM and TEE, adds an overhead on the computing time to generate the attestation. The paper [2] presents a detailed study of the performances of Histotrust according to the security level of the private key sk. On the one hand, on the blockchain side, a huge latency may be observed due to the time interval between two consecutive blocks. The delay between two blocks is very different from a blockchain to another. Ethereum implemented in private blockchain with *Clique* algorithm [3] as consensus protocol provides by default a time interval around 12 seconds between two consecutive blocks. As comparative example, two consecutive blocks are 10 minutes apart in the blockchain Bitcoin. On the other hand, the rate of data production by the real-time industrial app can be very high. To circumvent this problem, Histotrust uses the receipt that confirms the registration of an attestation in the ledger to trigger the read of a new data set coming from the industrial app.

5.3 Verification

The attestation history is available in the shared ledger and transparent to all stakeholders. It does not include confidential information, only cryptographic attestations enabling the verification. Each record is a transaction signed with sk, emitted from the account of the issuing device, and sent to the smart contract. It includes the fingerprint of the attested data set.

Two types of verifiers are distinguished:

- involved stakeholder: Any stakeholder is able to access the information present in the shared ledger. The registered attestations enable to authenticate the acting devices and their owner in a given time interval.
- independent auditor: An independent auditor, such as an insurance expert or a bailiff, may be accredited to request the raw data, to the authenticated device's owner, from the information registered in the shared ledger.

6 Embedded design

6.1 The IoT device: a System-on-Module

This section briefly presents the IoT platform design. A STM32MP157-EV1 evaluation board is associated with a STPM4RasPI TPM Expansion Board. The STM32MP157 is a single board computer composed of a dual-core ARMCortex-A7 core processor operating at 650Mhz forming a System-on-Module (SoM). The processor also integrates an ARM Cortex-M4 coprocessor, which makes it suitable for real-time tasks.

The dual-core ARM-Cortex-A7 is very low-power processor designed for smartphone or edge devices. It includes both a normal world operating with a Rich OS and a secure world with a TrustZone operating with OP-TEE OS. The transition from the normal world to the secure world is done by setting the NS bit in the SCR register to 1. The code executed remain confidential and is protected against logical attacks.

The coprocessor ARM-Cortex-M4 offers a real-time environment accessible from the normal world of the ARM-Cortex-A7 to extend its computing capabilities and increase its performances while preserving low-power consumption. The functions embedded in the ARM-Cortex-M4 are built upon the dedicated Hardware Architecture Layer (HAL). STMicroelectronics provides a protocol called RPMSG [21] to ensure the communication between the ARM-Cortex-A7micro-processor and the ARM-Cortex-M4 micro-controller.

The daughter board STPM4RasPI completes the STM32MP157 with a TPM 2.0 from STMicrolectronics. This board is connected through the GPIO making the TPM accessible from the OP-TEE environment via the SPI bus. An Ethernet connection and a serial link enable the monitoring of the SoM. A small screen displays some information about the hardware configuration.

6.2 Secure boot and measurement

The ARM-Cortex-A7 includes an open source Trusted Execution Environment (OP-TEE) implementing the ARM TrustZone technology. At start, a secure boot process is achieved according the application note [22] relying on Brainpool 256 ECDSA key. At start and during the execution in production mode, the integrity of the two embedded trusted apps is checked through measurement process. To enable this, the fingerprint of the apps binary code is previously provisioned in the TPM PCR registry as explained in paragraph 5.1.2.

6.3 Integration

The integration consists to make the industrial app and the attestation app working together in the SoM as depicted in the figure 5, while respecting the real-time constraint of the industrial app.

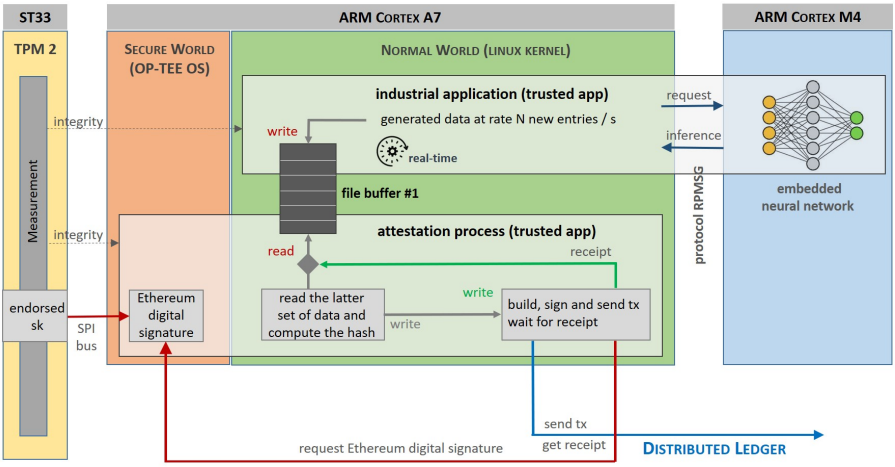


Fig. 5: Embedded design in the physical devices

The industrial app is embedded in the normal world operating on a linux kernel as rich OS of the ARM-Cortex-A7, with a part including the NN insulated in the ARM-Cortex-M4. It handles the pictures coming from the attached camera in the ARM-Cortex-A7. The pictures and transmitted to the input of the NN in the ARMCortex-M4, to request an inference. As output, the NN provides 10 heuristics, one by digit from '0' to '9'. The heuristics are carried to the ARM-Cortex-A7. Generally, the recognised digit corresponds to the highest probability.

The communication protocol between the ARM-Cortex-A7 and the ARM-Cortex-M4 microcontrollers is suggested by STMicroelectronics in [21]. It implements a virtual interface, noted *ttyRPMSG*, that enables the exchange of small size messages and low data flows. The transmission of small images to the ARM-Cortex-M4 with this protocol leads to a loss of information because the throughput is not sufficient. That why, Histotrust implements a new communication scheme between the ARM-Cortex-A7 and the ARM-Cortex-M4 on the SoM. The virtual interface *ttyRPMSG* is used to notify the presence of data in a shared memory, accessible to both microcontrollers, and the direction of the communication.

Several buffers are implemented in the shared memory in order to handle full duplex communications without loss of data. The data to attest composes the new entry written in the file 1. For the use-case considered, the format of each new entry is as follow:

$[index \| timestamp \| url \| hash \| inference]$

The field *url* is a pointer to the raw data in entry of the NN, while the field *hash* is the hash of the raw data. The field *inference* is composed of the 10 values of heuristic, one for each digit from '0' to '9'. Each heuristic is a floating value coding a probability between 0 and 1.

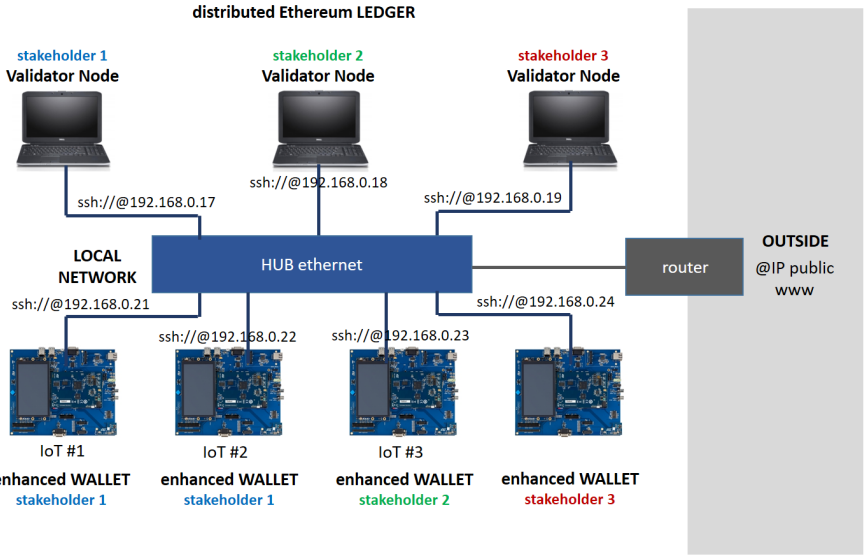


Fig. 6: network architecture

The industrial app writes in real-time in the file #1 all the data produced that needs to be attested. The size of this buffer is not limited, as it is stored on the SD card with several GB available. Only the industrial app is authorized to write in this file, while attestation app is authorized to read the file #1. The receipt received from the blockchain confirms the registration of the attestation of the previous data set in the ledger. This receipt triggers the read of the next data set in the file #1. The file #1 is stored in persistent memory. If a power failure occurs, the data is saved and the attestation process resumes where it left off when the power returns. The file #1 may be ex-filtrated by its owner.

The attestation app includes a part located in the normal world and another part located in the secure world of the ARM-Cortex-A7. The ST33 TPM is accessed from the secure world, thanks to the integration of the SYS layer of the TPM stack in the OP-TEE environment. The lightweight library mbedTLS is also embedded in the OP-TEE environment, providing cryptographic primitives and build custom functions such as the ethereum digital signature. In the normal world, low level commands enable the connection of the device with the remote blockchain through JSON-RPC convention.

6.4 Deployment

All the devices are distributed on a local network following a star topology around an access point. A proxy enables the communication with the outside to enable raw data ex-filtration. A consortium ethereum blockchain is locally deployed. Each stakeholder involved in the use case owns a validator node

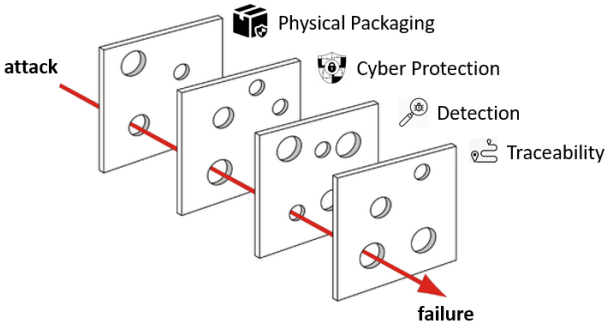


Fig. 7: Security model inspired from the swiss cheese model

with a complete copy of the ledger, and has one vote in the consensus protocol. The validator nodes are depicted with a computer in the figure 6. Thus, the governance of the system is ensured with equity and fairness by all the stakeholders.

The devices acting in the production line, are provided with the embedded apps, enabling to send transactions to the validator nodes. So, each device is the root-of-trust of the data it produces, forming a distributed root-of-trust network. The provisioning is done, independently by each device's owner, prior to the deployment of the hardware in the factory. The management of the access rights and authorizations is done through smart contracts.

7 Security analysis

7.1 The security model

In 1990, Reason [23] introduces the swiss cheese model to analyse the causality of an incident and manage risks. The physical device, that embeds the NN, integrates several security layers to protect and detect attacks or malfunctioning, as depicted in the figure 7.

The first layer is a physical protection that prevents access to the components embedded in the smart robot, and that remains physically damaged in case of intrusion. By this way, succeeding in a physical attack on the electronic components that support the NN is difficult and leaves marks. The second layer is the cyber protection against logical attacks. The use of secure hardware components such as TPM and OP-TEE to protect the cryptocraphic keys and seeds, is the foundation of this protection. The third layer is the detection of intrusions or tampering. At this layer, secure boot and measurement are deployed to monitor the integrity of embedded firmware and software. The fourth layer concerns the traceability to be able to understand what happens when the previous layers are bypassed. A blockchain is used to register the traces as attestations of the logged data produced by the embedded apps.

7.2 The asset

The assets to protect are the business-relevant data of the stakeholders. It is the logged data including all the relevant data produced by the physical devices, which contributes to make decisions of the digital command sent to the actuators. This includes inferences produced by the embedded NN (see figure 8). The authenticity should be ensured, as well as integrity and completeness.

The traceability is a valuable service to understand the origin and sequence of the events, while the raw data produced remains confidential to its owner. In order to reduce the attack surface on the electronic board, the different protection layers of the figure 7 integrate several countermeasures. The goal is to fulfil these security requirements:

- *R1: AI explainability*: The behavior of the embedded AI should be explainable.
- *R2: forward integrity*: The data attestation history must be immutable and transparent to the stakeholders. The raw data must be persistent and of integrity.
- *R3: public authentication*: Any stakeholder should be able to authenticate the devices issuing data in a given time interval through the attestations history.
- *R4: power failure*: No raw data or attestations should be lost in the event of a power failure.
- *R5: privacy-preserving data*: The raw data shall not be exposed to the other devices.
- *R6: verifiability*: An accredited auditor must be able to verify the data attestations.
- *R7: multiple stakeholders*: The scheme shall support multiple-stakeholders owning multiple devices issuing data concurrently.

7.3 The threat

The threat events are the tampering of the data produced, the production of fake or dysfunctional data, the spoofing of data or issuing devices, the theft of data.

The main sources of risks come from the following profiles:

- *Negligence*: this threat arises from unintentional human error, but causing a failure,
- *Ransacking*: this threat corresponds to a malicious action with the intention to destroy, tamper, spoof, modify value data,
- *Concurrence*: this threat may seek to destroy data like the ransacker, but also to steal valuable data for analysis.

The main stakeholders, involved in the smart manufacturing use case are:

- the *provider* of the smart robot, by default he is the owner of the logged raw data produced by its devices,

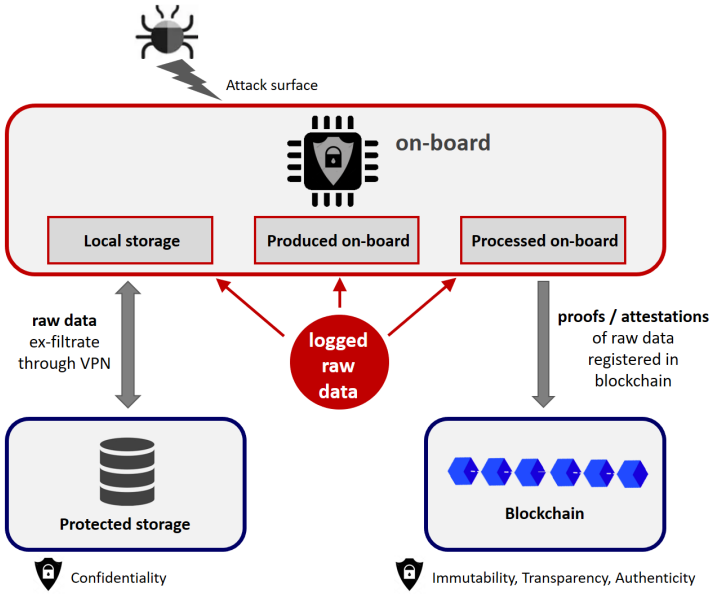


Fig. 8: Security of the embedded scheme

- the *expert* who learns the embedded AI,
- the *manufacturer* of the product (e.g. the car) for which the robot performs tasks,
- the *operator* of the smart robot during production,
- the *maintenance agent* who intervenes on the smart robot,
- the accredited and independent *auditor* mandated in case of litigation.

The table 1 shows the role that each stakeholder can play.

Table 1: possible profile of the various stakeholders

stakeholder	negligent		concurrent
provider	✓	✓	✓
expert	✓		
manufacturer	✓	✓	✓
operator	✓	✓	
maintenance agent	✓	✓	
auditor			

The provider of the smart robot may be negligent in providing an unreliable device, poorly configured, or in which bugs remain. In the event of a litigation, he must provide the integrity of the data requested by the auditor. Thus, it is the provider's responsibility to maintain the tamper-resistance and confidentiality of his data. As there are usually several suppliers of smart robots in a factory, they are potentially concurrent.

This may be an incentive to obtain confidential data from their concurrent for analysis to gain market share. The expert is responsible for the learning of the AI and the decision of the embedded NN. He must be able to explain how the heuristics are derived. The manufacturer is physically present in the factory and has access to the smart robots. He may take any profile of attacker in order to hide a problem for which he is responsible and pass the blame on to another stakeholder. An operator or a maintenance agent may make a human error, and possibly seek to cover it up by destroying elements.

The auditor's mandate is in the legal field, which gives him legal accreditation and independence from other stakeholders.

7.4 Security and Privacy review

R1: AI explainability. Explaining the behaviour of an AI requires measures to be implemented at the design stage. The blockchain technology provides obviously and by design the property of traceability. However, the blockchain does not manage the confidentiality of the traced data. This is why Histotrust proposes a scheme combining the use of a blockchain to transparently guarantee the properties of immutability, authenticity and ordering, and the use of private storage of raw data, under the responsibility of their owner.

R2: forward integrity. The blockchain ensures by design the forward integrity of the information recorded in the ledger. The ledger maintains the history of cryptographic attestations, each one being a pointer to a raw data set stored outside the blockchain. Thus, any tampering or removal of raw data is detectable.

R3: public authentication. The recorded attestation authenticates the device issuer,and all genuine device is endorsed by its owner. The consultation of the ledger allows any stakeholder to know the devices acting in a given time interval, and the order of the performed actions.

R4: power failure. Resilience when a power failure occurs, implies that no raw data or cryptographic attestations are lost. The use of a file buffer stored in permanent memory ensures data persistence in case of power failure.

R5: privacy-preserving data. This requirement covers raw data at storage and during transportation. The physical protection of the device in the factory make access to the board peripherals difficult and detectable. The ex-filtration of the raw data is performed through VPN.

R6: verifiability. Histotrust distinguishes two roles of verifiers. All the stakeholders can play the first role, having access to the attestations recorded in the ledger. The second role is reserved an accredited auditor, under a legal mandate, to request the raw data.

R7: multiple stakeholders. HistoTrust brings a solution where the number of stakeholders is not limited by using blockchain technology as a complement to existing technologies. The stakeholders ensure the governance together, each having a validator node.

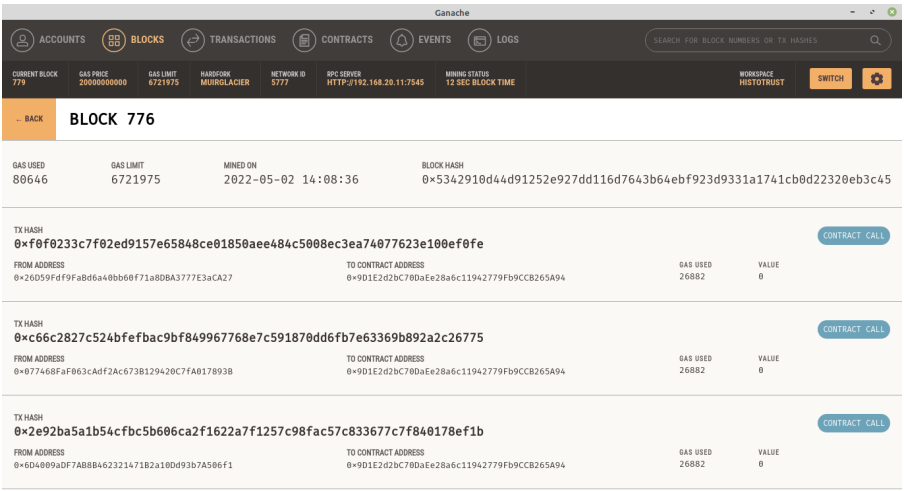


Fig. 9: Transactions included in one block

8 Audit

The audit is launched when an incident occurs. The goal of the audit is to determine the cause and the accountabilities with the maximum of transparency for the involved stakeholders. The audit takes place in two phases: the first to trace the events in a given time interval before the incident. The second is to analyse the behavior of the AIs involved.

8.1 Traceability of the events

The blockchain provides an immutable history, shared among all stakeholders, of all past events. The figure 9 shows an extract of the ledger securing a succession of blocks including the attestation history. The attestations are kept ordered, timestamped and of integrity.

Each block includes a tree of recorded transactions, as shown in the figure 10. The *sender address* authenticates the issuer device, while the *contract address* authenticates the recipient smart contract. The field *data* includes the fingerprint of the raw data set produced by the issuing device at the given time, whereas the field *gas* indicates the computing power required to execute the targeted smart contract in the blockchain. This value is an indicator of the energy consumed to execute an instance of the smart contract.

Until the request of personal data, any stakeholder member of the ecosystem can achieve the verification. The first step consists to get the recorded attestations of the considered time interval in the shared ledger. Each attestation authenticates the issuer device, as well as the owner who has endorsed his devices.

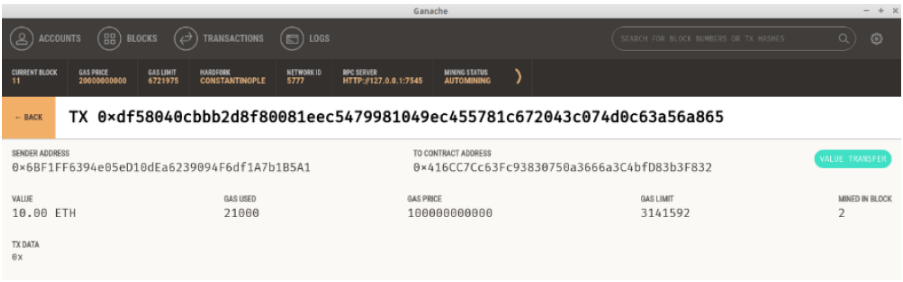


Fig. 10: Detail of an attestation registered in the blockchain

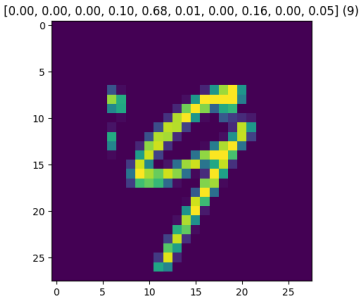


Fig. 11: example of an inference requiring explanation

8.2 Explainability of the AI

Each device's owner may be requested to provide the raw data associated to the recorded attestations. As these data are confidential, only an accredited and independent auditor is authorized to do this task regulated by the legal.

The provided data must be complete and of integrity, otherwise the accountability of the owner is engaged, with the suspicion of hiding a fraud. Each owner is responsible for keeping and protecting its logged raw data.

Once the completeness and the integrity of the attested data established, the analysis of the raw data is lead, in particular the analysis of the AI behavior. Each owner is responsible for providing an explanation of the behavior of its embedded NN.

At this stage, the analysis relies on tools and methods the expert used to explain the behavior of the NN, and on human expertise. For example, the picture presented in the figure 16 is labelled '9'. However, the inference from the embedded NN recognises the digit '4' with a probability of 68%, the digit '7' at 16% and the digit '9' at 5%. With a school case and a labelled image, one knows that it's a '9'. But the pictures acquired by the smart robot's onboard cameras are not labelled. And, only the explainability of the learning model and human expertise can remove the doubt on the most likely pattern. In a factory, the smart robot are supervised by human operators. So, one can

consider that if the inference does not return any heuristics above a certain threshold, e.g. 71%, the decision is the accountability of the human operator. On the other hand, when the error is obvious, for example, the NN recognises a '3' with 95% certainty when it is a '0', the human operator will not be solicited, and potentially this can lead to an incident on the production line. This may be due to an adversarial attack, i.e. an attack on the NN affecting the cyber protection layer (see figure 11) and not detected by the embedded system. The traceability implemented with HistoTrust allows to discover the cause.

9 Conclusion

This paper introduces Histotrust, a robust scheme using TEE and TPM secure components to trace the behavior of embedded AI. It begins with the challenge of embedding a learnt NN in an ARM-Cortex-M4 microcontroller. Next, based on an attestation scheme to an ethereum ledger, an embedded design is proposed to secure the NN, ensure its robustness and enable the explainability of its behavior. Then, several devices, following a distributed architecture, are deployed around a blockchain. The security analysis and the audit process provides verification tools that brings trust and fairness between the stakeholders involved in the use case. In future work, the privacy preserving data will be deepen, and some cryptographic process will be ported to the TPM.

Acknowledgments. This work is a collaborative research action that is partially supported by (CEA-Leti) the European project ECSEL InSecTT [2] and by the French National Research Agency (ANR) in the framework of the Investissements d'avenir program (ANR-10-AIRT-05, irtnanoelec)

References

[1] P. Jonathon Phillips, Carina Hahn, Peter Fontana, Amy Yates, Kristen K. Greene, David A. Broniatowski, Mark A. Przybocki, "Four Principles of Explainable Artificial Intelligence", NIST Interagency/Internal Report (NISTIR) - 8312, 2021.

[2] Dylan Paulin, Christine Hennebert, Thibault Franco-Rondisson, Romain Jayles, Thomas Loubier, Raphaël Collado, "HistoTrust: ethereum-based attestation of a data history built with OP-TEE and TPM", In proceedings of the 14th International Symposium on Foundations Practice of Security, 2021.

[2]www.insectt.eu, InSecTT: ECSEL Joint Undertaking (JU) under grant agreement No 876038. The JU receives support from the European Union's Horizon 2020 research and innovation program and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, Turkey. The document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains.

[3] Péter Szilágyi, "EIP-225: Clique proof-of-authority consensus protocol", Ethereum Improvement Proposal, https://eips.ethereum.org/EIPS/eip-225

[4] Thomas Hardjono and Ned Smith, "An attestation architecture for Blockchain networks", arXiv:2005.04293 [cs.CR], 2020.

[5] Dhiman Chakraborty, Lucjan Hanzlik and Sven Bugiel, "simTPM: User-centric TPM for Mobile Devices", In Proceedings of the 28th Conference USENIX Security Symposium, SSYM'19, USENIX Association, , pp. 533-550, 2019, isbn: 978-1-939133-06-9.

[6] Christine Hennebert and Florian Barrois, "Is the blockchain a relevant technology for the industry 4.0?", In Proceedings of the 2nd IEEE Conference on Blockchain Research & Applications for Innovative Networks and Services, BRAINS'20, pp. 212-216, IEEE Publisher, 2020, doi: 10.1109/BRAINS49436.2020.9223290.

[7] Carlton Shepherd, Raja Akram and Konstantinos Markantonakis, "EmLog: Tamper-Resistant System Logging for Constrained Devices with TEEs", In Proceedings of the 11th IFIP International Conference on Information Security Theory and Practice, WISTP'17, pp. 75-92, Springer International Publishing, 2017, doi: 10.1007/978-3-319-93524-9_5.

[8] Carlton Shepherd, Ghada Arfaoui, Iakovos Gurulian, Robert P. Lee, Konstantinos Markantonakis, Raja Naeem Akram, Damien Sauveron and Emmanuel Conchon, "Secure and Trusted Execution: Past, Present, and Future - A Critical Review in the Context of the Internet of Things and Cyber-Physical Systems", In Proceedings of the IEEE Trustcom/BigDataSE/ISPA, pp. 168-177, IEEE Publisher, 2016, doi: 10.1109/TrustCom.2016.0060.

[9] Bruce Schneier and John Kelsey, "Cryptographic support for secure logs on untrusted machines", In Proceedings of the 7th Conference on USENIX Security Symposium, Volume 7, SSYM'98, USENIX Association, 1998.

[10] Vishal Karande, Erick Bauman, Zhiqiang Lin, Latifur Khan, "SGX-Log: Securing System Logs With SGX", In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS '17, pp. 19–30, ACM Publisher, 2017.

[11] Louis Shekhtman and Erez Waisbard, "EngraveChain: Tamper-proof distributed log system", In Proceedings of the 2nd Workshop on Blockchain-enabled Networked Sensor, BlockSys'19, ACM Publisher, 2019, doi: 10.1145/3362744.3363346.

[12] Uzair Javaid, Muhammad Naveed Aman and Biplab Sikdar, "BlockPro: Blockchain based Data Provenance and Integrity for Secure IoT Environments", In The 1st Workshop on Blockchain-enabled Networked Sensor Systems, BlockSys'18, ACM Publisher, 2018, doi: 10.1145/3282278.3282281.

[13] Atis Elsts, Efstathios Mitskas and George Oikonomou, 2018, "Distributed Ledger Technology and the Internet of Things: A Feasibility Study", In The 1st Workshop on Blockchain-enabled Networked Sensor Systems, BlockSys'18, ACM Publisher, 2018, doi: 10.1145/3282278.3282280.

[14] Trusted Computing Group, "TCG Trusted Attestation Protocol (TAP) Use Cases for TPM Families 1.2 and 2.0 and DICE", Version 1.0, Revision 0.35, 2019.

[15] George Coker, Joshua Guttman, Peter Loscocco, Amy Herzog, Jonathan Millen, Brian O'Hanlon, John Ramsdell, Ariel Segall, Justin Sheehy, Brian Sniffen, "Principles of remote attestation", International Journal of Information Security, Volume 10, pp. 63–81, Springer, 2011, doi: 10.1007/s10207-011-0124-7.

[16] Kang Yang, Liqun Chen, Zhenfeng Zhang, Christopher J.P. Newton, Bo Yang and Li Xi, "Direct Anonymous Attestation with Optimal TPM Signing Efficiency", eprint 1128, 2018.

[17] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, K. R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications", in Proceedings of the IEEE, 2021, 109(3), 247-278.

[18] - Y. LeCun et al., "Backpropagation Applied to Handwritten Zip Code Recognition," in Neural Computation, vol. 1, no. 4, pp. 541-551, Dec. 1989, doi: 10.1162/neco.1989.1.4.541.

[19] R. Mishra, H. P. Gupta, T. Dutta, "A survey on deep neural network compression: Challenges, overview, and solutions". 2020, arXiv preprint arXiv:2010.03954.

[20] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean,... X. Zheng, 2016"TensorFlow: A System for Large-Scale Machine Learning", In 12th USENIX symposium on operating systems design and implementation (OSDI 16), 2016, pp. 265-283.

[21] STMicroelectronics, "Linux RPMsg framework overview", https://wiki.st.com/stm32mpu/wiki/Linux_RPMsg_framework_overview

[22] STMicroelectronics, "STM32MP15ROM code secure boot", https://wiki.st.com/stm32mpu/wiki/STM32MP15_ROM_code_secure_boot

[23] James Reason, "The Contribution of Latent Human Failures to the Breakdown of Complex Systems", Philosophical Transactions of the Royal Society of London, Series B, Biological Sciences, vol. 327, n°1241, 1990, pp. 475–484, doi: 10.1098/rstb.1990.0090.